Applicant : William R. Wheeler et al.
Serial No. : 09/942,116
Filed : August 29, 2001
Page : 2 of 13

Attorney's Docket No.: 10559-601001 / P12885

Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

<u>Listing of Claims</u>:

1.   (Currently Amended) A computer implemented method of simulating a logic design, the method comprising:

storing a first state to identify in a simulation of a logic design whether a node included in the logic design has a logic high value;

storing a second state to identify in simulation of the logic design whether the node has a logic low value;

storing a third state to identify in simulation of the logic design whether the node has an undefined state; and

performing a three state simulation of the logic design to determine an output of the node in simulation of the logic design based on the first state, the second state, and the third state; and

determining if the <u>three state</u> simulation of the logic design was successful, based on whether the output of the node has an undefined state<u>, the three state simulation being successful if the output of the node is a defined state</u>; and

performing a four state simulation of the logic design if the three state simulation of the logic design was successful.

2.   (Previously Presented) The computer implemented method of claim 1 further comprising determining whether the output of the node in simulation of the logic design has a high impedance state based on the first state, the second state, and the third state.

3.   (Previously Presented) The computer implemented method of claim 1 further comprising storing the third state only if at least two sources drive the node.

4.   (Canceled)

5.   (Previously Presented) The computer implemented method of claim 1 further comprising:

determining the output based on which of the first state, the second state, and the third state is at a logic high.

6.   (Currently Amended) An article for simulating a logic design, the article comprising:

a machine readable medium which contains machine executable instructions, the instructions causing a machine to:

store a first state to identify in a simulation of a logic design whether a first node included in the logic design has a logic high value;

store a second state to identify in simulation of the logic design whether the first

node has a logic low value;

store a third state to identify in simulation of the logic design whether the first

node has an undefined state; and

perform a three state simulation of the logic design to determine an output of the node in

simulation of the logic design based on the first state, the second state, and the third state; and

determineif the three state simulation of the logic design was successful, based on

whether the output of the node has an undefined state, the three state simulation being successful

if the output of the node is a defined state; and

perform a four state simulation of the logic design if the three state simulation of the logic

design was successful.

7.   (Original) The article of claim 6 further causing a machine to determine whether

the output of the node in simulation of the logic design has a high impedance state based on the

first state, the second state, and the third state.

8.   (Original) The article of claim 6 further causing a machine to store the third state

only if at least two sources drive the node.

9.   (Canceled)

Applicant : William R. Wheeler et al.   Attorney's Docket No.: 10559-601001 / P12885
Serial No. : 09/942,116
Filed   : August 29, 2001
Page   : 5 of 13

10.   (Original) The article of claim 6 further causing a machine to:

determine the output based on which of the first state, the second state, and the third state

is at a logic high.

11.   (Currently Amended) An apparatus for simulating a logic design, the apparatus

comprising:

a memory that stores executable instructions; and

a processor that executes the instructions to:

store a first state to identify in a simulation of a logic design whether a

first node included in the logic design has a logic high value,

store a second state to identify in simulation of the logic design whether

the first node has a logic low value,

store a third state to identify in simulation of the logic design whether the

first node has an undefined state, and

perform a three state simulation of the logic design to determine an output

of the node in simulation of the logic design based on the first state, the second

state, and the third state; and

determine if the _three state_ simulation of the logic design was successful,

based on whether the output of the node has an undefined state_, the three state_

simulation being successful if the output of the node is a defined state; and

perform a four state simulation of the logic design if the three state

simulation of the logic design was successful.

12.   (Original) The apparatus of claim 11 further executing the instructions to

determine whether the output of the node in simulation of the logic design has a high impedance

state based on the first state, the second state, and the third state.

13.   (Original) The apparatus of claim 11 further executing the instructions to store

the third state only if at least two sources drive the node.

14.   (Original) The apparatus of claim 11 further executing the instructions to perform

four state simulation of the logic design subsequent to the simulation of the logic design if the

simulation of the logic design was successful.

15.   (Original) The apparatus of claim 11 further executing the instructions to

determine the output based on which of the first state, the second state, and the third state is at a

logic high.

16.   (Previously Presented) A method of simulating a logic design, the method

comprising:

Applicant : William R. Wheeler et al.   Attorney's Docket No.: 10559-601001 / P12885
Serial No. : 09/942,116
Filed   : August 29, 2001
Page   : 7 of 13

storing three bits of state information for a node included in a logic design, where the

state information is represented by one of two possible values of the three bits of state

information as a first bit of the three bits representing the presence or absence of a high state for

the node, a second bit of the three bits representing the presence or absence of a logic low state

for the node, and a third bit of the three bits representing the presence or absence of an undefined

state based for the node; and

checking states of the three bits in simulating operation of the logic design.

17.   (Original) The method of claim 16 further comprising determining an output of

the node in simulation of the logic design based on the three bits of state information.

18.   (Original) The method of claim 16 further comprising determining whether an

output of the node in simulation of the logic design has a logic high state, a logic low state, or an

undefined state based the three bits of state information.

19.   (Original) The method of claim 16 further comprising determining whether an

output of the node in simulation of the logic design has a high impedance state based the three

bits of state information.

20.   (Original) The method of claim 16 further comprising storing four bits of state

information for the node included in the logic design;

checking the four bits in a second simulating operation of the logic design; and

determining whether an output of the node in simulation of the logic design has a logic

high state, a logic low state, an undefined state, or a high impedance state based the four bits of

state information.

21.   (Original) The method of claim 16 further comprising determining an output of

the node based on which of the three bits of information is at a logic high.

22.   (Previously Presented) A method for simulating a logic design using cycle based

simulation, the method comprising:

attempting to write a result of a logic computation instruction to a location in a first

memory page storing an original value, the memory page being write protected;

copying the instruction to a second memory page;

executing the second memory page starting with the instruction;

unprotecting the write protected memory page and storing a result of the instruction at the

location in the first memory page;

rewriting the original value to the location in the first memory page; and

re-protecting the write protected memory page.

Applicant : William R. Wheeler et al.
Serial No. : 09/942,116
Filed   : August 29, 2001
Page   : 9 of 13

Attorney's Docket No.: 10559-601001 / P12885

23.   (Original) The method of claim 22 further comprising inserting an illegal instruction after the instruction in the second memory page that triggers the rewriting and the re protecting.

24.   (Original) The method of claim 22 further comprising generating an exception if a computer system attempts to write the result of the logic computation instruction to the location in the write protected memory page